COMPUTER RECREATIONS

An expert system outperforms mere mortals as it conquers the feared Dungeons of Doom

by A. K. Dewdney

Every year thousands of people die a fantasy death in the Dungeons of Doom while playing Rogue, one of a new generation of computer adventure games. The player watches a map of the dungeons on the display screen and directs the actions of his or her game persona, a character called the Rogue. The object of the game is to descend through the 26 levels of the dungeons, seize the Amulet of Yendor and return safely to the surface, gathering gold and slaying or escaping from monsters along the way. Few human players survive the dangers of this subterranean odyssey, let alone return with the amulet.

The fantasy can be quite vivid; one quickly leaves awareness of keyboard and display screen far behind. As the Rogue, I approached the entrance to the dungeons with some trepidation: it was night and the ancient ruins that marked the point of my imminent descent had a gloomy and forbidding appearance. In preparation I set out my enchanted mace, a bow and a quiver of arrows snatched from a dragon's hoard in the distant Dark Mountains. I donned my elf-crafted armor, picked up my weapons and food and stepped into the stygian darkness of a stairway.

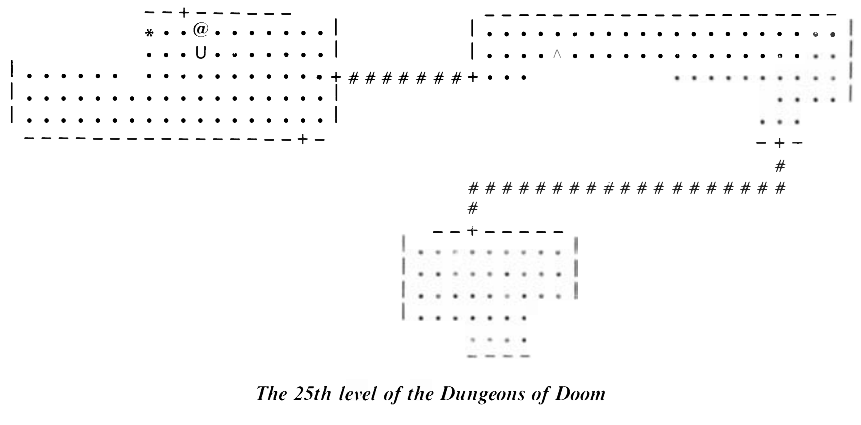
Just as the descent began to seem endless, I bumped into an oaken door and swung it cautiously open. There before me was the first room on the uppermost level of the Dungeons of Doom. It was dimly lighted by tapers and I trod carefully to the middle of the chamber, the better to survey it. Suddenly the floor under me gave way and for a breathless second I was falling. With a sickening thud I landed in another room. This one was much darker and, even when my eyes became accustomed to the dark, I could see only a few feet in any direction. To make matters worse, I could hear something moving about in this chamber. Nauseous with fear, I stumbled to my feet to find myself confronting a squat, horrid armor-clad figure holding a club. As it raised its weapon to strike, a rush of adrenalin cleared my head. I fitted an arrow to my bow, drew it and fired all in one fluid motion. (Fortunately I took archery as a noncredit course in my senior year.) There was a swish, a thwock and a squeal as the goblin (for that is what the creature was) fell to the floor quivering with unvented rage. I stepped gingerly away from it, determined to find a stairway up and out of the Dungeons of Doom. I thought of my cozy home and my writing desk with the unfinished "Computer Recreations" article on it.

As I shuffled slowly in the direction in which I guessed a stairway to lie, my boot struck what felt like a small heap of stones. I looked down. Even in the dark something gleamed. Gold! It occurred to me to explore just one more room before returning to the surface....

Aside from the intensity of vicarious experience, Rogue goes a step beyond traditional adventure games in at least two respects. First, the layout of the terrain is generated by the program itself and the layout changes from one game to the next. Second, Rogue supplies the human player with a plan view of the current level of dungeons as explored so far [*see illustration on this page*]. Various features are indicated by different keyboard characters drawn on the screen. For example, bars and hyphens represent walls, number and percent signs represent passages and stairways respectively, plus signs represent entrances and carets represent traps. Objects and features of the chamber may or may not be visible depending on whether or not the chamber is illuminated. When the Rogue enters this particular level, no features are shown on the screen. In order for them to be represented the Rogue (represented by an "at" sign) must discover them. And so he explores, stumbles on some traps, scouts parts of various rooms and traverses passageways. He may encounter the letter *U,* which represents a monster known as the Umber Hulk.

Below the character-oriented graphic plan the screen displays the Rogue's current statistics. In the illustration the Rogue is on the 25th level, he has accumulated 7,730 pieces of gold, he has earned 77 hit points (but only 25 remain because of damage sustained), he has accumulated strength to the 15th degree (18 is the highest), he wears class-9 armor and has 30,668 experience points, enough to place him on the exalted 13th order of experience. At the 25th level of the Dungeons of Doom this Rogue has only one more stairway to descend before reaching the bottom level and attempting to snatch the coveted Amulet of Yendor. The Rogue must first dispose of the Umber Hulk, however.

This discussion explores only the surface of the game. It would be possible to write an entire book of rules and advice for playing Rogue. So far the only available document of that nature is a terse, eight-page report written by Michael C. Toy and Kenneth C. R. C. Arnold, *A Guide to the Dungeons of Doom* [see "Bibliography," page 128]. Unfortunately it is available only to users of the VAX/UNIX time-sharing system. Arnold tells me that a version of Rogue suitable for the IBM PC is now available from A.I.



The 25th level of the Dungeons of Doom

Design, 201 San Antonio Circle, Suite 115, Mountain View, Calif. 94040. Toy and Arnold, Rogue's creators, evidently think the best way to learn the game is to play it. Nevertheless, the general features of the game can be delineated.

@	the Rogue	h	move one space west
–|	walls of rooms	j	move one space south
+	doorway	k	move one space north
#	passage between rooms	l	move one space east
·	floor of room	>	descend stairway
%	staircase	<	ascend stairway
∧	trap	m	move onto something
)	weapon	s	search for trap
]	piece of armor	∧	identify trap
*	gold	. /e	rest/eat
!	flask of potion	W/T	wear/take off armor
?	magic scroll	P/R	put on/remove ring
:	food	d	drop an object
/	magic wand	q	quaff a potion
=	magic ring	r	read a scroll
A–Z	uppercase letters	z	zap with magic wand
	denote denizens	t	throw an object
	of dungeons	w	wield a weapon
		f	fight to the death

Map symbols (**left**) *and commands for the game of Rogue*

At each level the playing area is divided into squares. The Rogue occupies one square at a time as he explores his vicinity. Movement within rooms and passages is controlled by typed letter commands such as *h, j, k* and *l* that move the Rogue a single square in one of the four main directions. Other commands produce diagonal movement or continued motion in a given direction. To make the Rogue ascend or descend a stairway, the player must type the character < or >.

When an object is discovered near the Rogue's current location, moving the Rogue to the square containing the object automatically causes the object to be picked up. If the player wants to move onto that square without picking up the object, he or she must type *m* followed by the appropriate direction character. It is also possible to have the Rogue search for traps by typing *s*, an action that effects a search of neighboring squares. A caution is in order: this search has only a 20 percent chance of discovering a trap.

Occasionally the Rogue must rest (type a period) or eat (type an *e*) to regenerate strength expended in wandering about or in subduing monsters. Apart from a limited amount of food carried in a backpack, the Rogue has nothing else to eat except what can be found on the dungeon floors (unappetizing as this may sound).

When a piece of armor is found, the Rogue may pick it up (automatically placing it in the backpack) or wear it (type a *W*). Armor naturally gives the Rogue more protection in combat, but armor may be cursed. If the armor is cursed, efforts to take it off (type a *T*) are useless without a magic scroll that breaks the curse. Any magic rings found by the Rogue may be put on (type a *P*) or removed (type an *R*) unless they are cursed.

At times it becomes necessary for the Rogue to drop a flask and a scroll or two (type a *d* followed by an object character) because his pack is full. (A Rogue in full pack cannot pick up armor.) Before dropping a flask, however, the Rogue might benefit from drinking the unknown contents: some potions have a healing effect and others enable the Rogue to see the Invisible Monster. On the other hand, the potion may merely cause confusion, so that if one types a command for the Rogue to move north, he may wander off in a random direction. By the same token, there may be an advantage to reading a scroll (type an *r*) before discarding it: certain formulas remove the curse from one's armor.

The most impressive magic devices at the Rogue's disposal are wands. Depending on the type of wand picked up, the Rogue may zap a monster (type a *z*) with various effects: he may transport it to a random location, shoot fireballs at it or change it into another monster. The last type of wand, called a polymorph wand, is best used on the more horrible monsters: it is much better to transform the dread Purple Worm into a bat than vice versa.

When the Rogue discovers a monster, a good policy might be to leave it alone. Sometimes a monster is sleeping and will not attack if left undisturbed. If a fight seems inevitable, however, one may wield a weapon (type a *w*) and then fight to the death (type an *f*). This can be done only after one has moved the Rogue next to the monster. The outcome of the struggle will depend probabilistically on such factors as the Rogue's current strength level, degree of experience and armor class.

Foremost in the ranks of players of Rogue stands a top-caliber nonhuman player. For the past four years a computer program has been playing the game and matching the prowess of the best Rogue runners in the quest for the amulet and gold; it provides as well an intriguing opportunity to watch an expert system at work.

On February 16, 1984, at the University of Texas at Austin, a Rogue manipulated by a program fought off all monsters, amassed a considerable pile of gold and returned with the amulet. Bearing the distinctly un-Tolkienesque name of ROG-O-MATIC, this program directed the Rogue's every step, every pause, every throw and blow.

ROG-O-MATIC is the creation of four graduate students in the Computer Science Department at Carnegie-Mellon University in Pittsburgh: Andrew Appel, Leonard Hamey, Guy Jacobson and Michael Mauldin. ROG-O-MATIC links programmed knowledge sources with expert systems to make decisions about what to do in every conceivable underground situation.

When a human plays Rogue, a stream of commands from the keyboard flows by way of the operating system into the ROGUE program. The program automatically decides which monster to confront the Rogue with next, which layout to use for the next level of dungeons, and so on. The program transmits this information, again by way of the operating system, to the display screen in order to keep the human informed of the current situation.

In replacing a human player, the ROG-O-MATIC program intercepts the keyboard character stream and transmits characters of its own to the ROGUE program. The latter has no idea (so to speak) that another program is playing. By the same token, information directed by the ROGUE program to the screen is also directed toward the ROG-O-MATIC program so that it can keep its own map of the Dungeons of Doom.

The ROG-O-MATIC program consists of 12,000 lines of the programming language C; it is even longer and more complicated than the ROGUE program. ROG-O-MATIC began in 1981 as what its original creators, Appel and Jacobson, thought would be a "simple project." Shortly after Mauldin joined the group the early Rogue-playing program went through an extensive round of modifications, each version adding some enhancement in tactical or strategic play. By the time the fourth member, Hamey, was contributing to program development, it was beginning to dawn on the authors that they had created, in effect, an expert system. Such systems constitute a key application of the so-called Fifth Generation of computers, projected to arrive in the marketplace in the late 1980's. An expert system is designed to embody and project human expertise in a variety of areas from medicine to engineering.

By using the type of software architecture that is suited to expert systems, the creators of ROG-O-MATIC were able to design and modify their program with relative ease. In particular, they organized the kinds of knowledge and expertise required by the Rogue into a hierarchy of various subsystems [*see illustration below*].

For example, one high-level expert (called Melee) controls fighting during combat and another high-level expert (called Target) directs the Rogue's pursuit of monsters. Both these experts use a lower-level expert called *battle,* which carries out special attacks or initiates retreat as the situation demands. The battle expert thus sometimes calls on the retreat expert for help, and the latter invariably draws on a source of knowledge called *pathc.* This is a special algorithm that searches the local terrain for the shortest path to whichever location is specified. Shortest-path algorithms are well developed in general, and in this particular application only the fastest algorithm possible is acceptable; it is used almost continuously as ROG-O-MATIC explores the Dungeons of Doom. Knowledge about terrain used by *pathc* comes from *termap,* essentially a data structure recording the terrain features so far discovered by the Rogue as he explores a particular level of the dungeons. Finally, *termap* obtains all its knowledge from *sense,* a low-level data structure containing all the relevant output of the ROGUE program.

Before listing the duties of other experts and knowledge sources, I should like to reexamine one of the experts, *battle,* in more detail. Once a battle is under way, the melee expert calls on the battle expert to decide whether to attack or to retreat. Discretion being the better part of valor, the battle expert first determines the desirability and feasibility of retreat. In order to do so it must check some preconditions:

1. The Rogue is not currently under the influence of the potion of confusion (which could cause the Rogue to flee directly toward the monster).

2. A monster is not already holding the Rogue fast.

3. It would be possible to die in one melee round (avoiding conflict would thus be highly desirable).

4. The retreat expert can find an escape route (a retreat is possible).

If all four preconditions are met, ROG-O-MATIC will turn the matter of the Rogue's retreat over to the retreat expert. If they are not met, the battle expert now runs through a list of aggressive possibilities.

1. If it is possible for the Rogue to die in one melee round, if the monster is nearby and visible to the Rogue and if the Rogue happens to have a teleportation wand, then point the wand at the monster.

2. If it is possible for the Rogue to die in one melee round, if the monster is nearby and if the Rogue has a teleportation scroll, then read the scroll's magic formula.

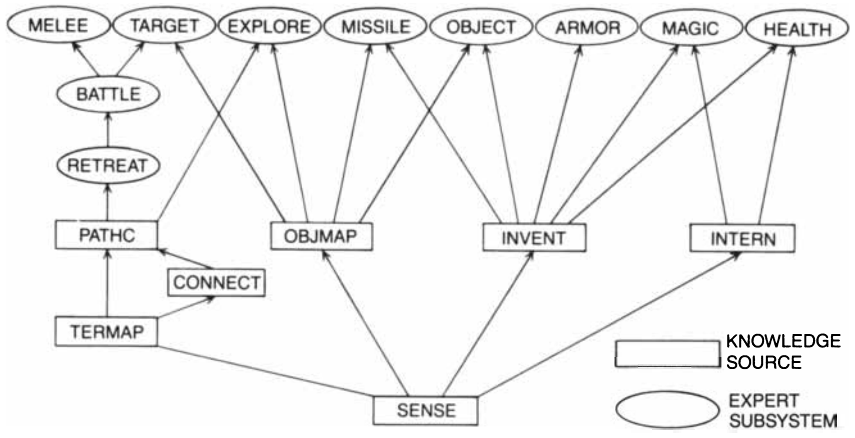
If neither of these conditions prevails in an encounter, there is no longer an alternative to going at least one round with the monster, and so the ROG-O-MATIC program cheerfully commits its Rogue persona to the battle. Perhaps the Rogue is strong enough to endure at least one melee round. Otherwise he may well be killed.

Other experts used by the program include an exploration expert (*explore*), which decides where to explore next and what motions to use. Also included are a missile expert, which handles the firing of arrows, rocks, spears and so on at monsters, an expert for choosing which objects to pick up, called *object,* an expert for choosing what armor to wear, an expert at using magic and an expert called *health,* which decides when to eat or rest. *Objmap* is a data structure that keeps track of the location and history of all objects so far encountered, *invent* is an inventory of items in the Rogue's backpack and *intern* is an internal-state recognizer that watches over the Rogue's readiness for exertion.

It is no surprise that ROG-O-MATIC can play a game of Rogue quickly. After a few minutes of computer time it is all over one way or the other. To date the program has played more than 12,000 games of Rogue at Carnegie-Mellon alone; statistics cited by ROG-O-MATIC's four creators tend to support the claim that the program's abilities now exceed those of the vast majority of human Rogue-playing experts. For example, in a test conducted at Carnegie-Mellon during a three-week period in 1983, ROG-O-MATIC had a higher median score than any of the 15 top Rogue players at the university. ROG-O-MATIC both outperforms most humans and shows some striking differences from them in its style of play. According to Mauldin, the program is careful and unimaginative. It explores efficiently and avoids all the redundant searches that humans are likely to undertake. Its fighting style, however, is rather methodical and, unless it is lucky, it misses some of the swashbuckling possibilities of human play.

Consider, for example, the factor of luck in its history-making performance last February at the University of Texas at Austin. After overcoming the incredible dangers of the deepest levels of the Dungeons of Doom, the Rogue (under ROG-O-MATIC direction, of course) found the Amulet of Yendor in a passageway on the 26th level. Racing toward the surface with his prize, the Rogue encountered on the 22nd level one of the worst dungeon monsters, a fire-breathing dragon. The Rogue drew his sword but the dragon exhaled a bolt of fire first. The searing flame missed the Rogue, who was standing in a doorway at the time, struck a wall and bounced directly back at the dragon, scorching it severely. With its remaining strength the dragon advanced toward the Rogue, who dispatched it with a sword blow. The Rogue then continued on to the surface, dealing with minor monsters as he went. When he emerged into the light of day, he had in his possession not only the amulet but also 6,913 pieces of gold and other objects.

The reader, if intrigued by the fore-



Expert subsystems and knowledge sources used by the program ROGUE going account, may want to sample the fun. For that purpose the following puzzle was constructed by Mauldin. In addition to being an introduction to the joys of playing Rogue, it illustrates the difficulty of building intelligence into the ROG-O-MATIC program.

A chamber on the 26th level of the Dungeons of Doom is currently inhabited by the Purple Worm (*P*), the Griffin (*G*) and the Rogue (@). The Amulet of Yendor (,) lies on the far side of a stairway (%) with respect to the Rogue.

```
 - - - - - - - - - - - - - - - - - -
| . . . . . . . . . . . . . . . . . |
| . . P . . . . . . . . , . . . . . |
| . . . . . . . . . . . . % . G . . |
| . . . . . . . . . . . . . @ . . . +
 - - - - - - - - - - - - - - - - - -
```

The Rogue's goal is to grab the amulet and race up the stairway without getting killed. It happens that the Rogue has only one hit point left and cannot risk a confrontation with either the Purple Worm or the Griffin. Fortunately the Purple Worm is fast asleep and the Rogue, wearing the Ring of Stealth, has only to avoid stepping on it in order not to wake it up. The Griffin, on the other hand, is very much awake and in pursuit of the Rogue. Although the Griffin would never disturb a fellow monster, it would like very much to occupy the same point as the Rogue, whom it would dismember with one slash of its cruel claws.

It is the Rogue's turn to move. Each move by either the Rogue or the Griffin involves occupying any of the eight adjacent points. So single-minded is the Griffin that it always chooses a move on the most direct path to its prey. If the Rogue makes it to the stairway, he is safe from both the Griffin and the Purple Worm: monster union rules forbid pursuit to other levels.

Last November's article on the Tower of Hanoi and the Chinese rings drew a variety of responses ranging from the mathematical to the metaphysical. Simple, nonrecursive solutions to the puzzles were presented by several readers, including Edward T. Price of Eugene, Ore. He suggests that Tower disks be painted alternately in two colors, say black and white, in order of increasing size. The puzzle is then solved quickly by adding the following rule to the ones originally given: Never place a disk on another disk of the same color. There is then no choice about where each one should be placed. The corresponding puzzle posed for the Chinese rings was solved by Morris S. Samberg of Howard Beach, N.Y. One alternates between moving (slipping on or off) the first ring and moving some other ring. Only one other ring may ever be moved. If the number of rings is odd, start with the first ring; otherwise start with the other one. Rob Hardy of Dayton, Ohio, summarizes this simple solution in the following verse:

An iterative solution
That's sure to spoil the fun:
Alternate changing the end ring
With changing some other one.

No reader was able to find a simple mathematical scheme behind the King Wen ordering of *I Ching* hexagrams. Many readers observed that hexagrams are paired across columns in a simple way, but this hardly explains the overall order. Homer E. Brown, an electrical-engineering consultant in Cary, N.C., produced a rather suggestive analysis, however: From each hexagram count up or down 10 hexagrams, skipping from the end of one column to the beginning of the next if necessary. The hexagram thus arrived at always has a simple relation to the one started with. The rule is not deterministic, however, as one never knows whether to count up or down.

In spite of this seeming relation (which has a number of exceptions in any event), I incline to the view that King Wen's arrangement follows metaphysical principles. Our attempt to project current preoccupations with science and technology into past cultures results in a distorted view of what once was. As a general rule the "scientific" systems that once existed served a role strictly subservient to a largely religious world view. Similar opinions are offered by Bernard X. Bovasso of Saugerties, N.Y. The *I Ching* is concerned with the ordering of time, maintains Bovasso, and so was King Wen. David White, a philosopher at Macalester College in St. Paul, Minn., referred to a translation of the *I Ching* by James Legge [see "Bibliography," page 128]. An appendix to this work lays out a metaphysic governing the order of all 64 hexagrams. For example, the first three are connected respectively with the concepts of heaven, earth and chaos. The chaos hexagram, denoting what is regarded as the disorder of all created things taken collectively, follows heaven and earth since all created things fill the space between heaven and earth.

Readers attempting to solve the 9-by-9 pursuit problem in the December column on sharks and fish may have become enmeshed in unnecessary complications. The ocean grid containing the two sharks and three fish should have been 9 by 10, therefore add one more vertical line to the grid.

Tina has never had a Teddy Bear.

A mother's love. A doll to cuddle. Tina knows nothing of these things. But she does know fear, rejection, and hunger.

For just $18 a month, you can help save a child like Tina.

Through our sponsorship program you can help provide a child with a better diet, clothes, medical attention, school. And even a toy or two.

But don't wait. There are so many. And somewhere, right now, a child is dying from starvation and neglect.

Write to: Mrs. Jeanne Clarke Wood
Children, Incorporated, P.O. Box 5381
Dept. SA2T5, Richmond, Va. 23220 USA

☐ I wish to sponsor a ☐ boy, ☐ girl, in ☐ Asia, ☐ Latin America, ☐ Middle East, ☐ Africa, ☐ USA, ☐ Greatest Need.
☐ I will give $18 a month ($216 a year). Enclosed in my gift for a full year ☐ , the first month ☐. Please send me the child's name, story, address and picture.
☐ I can't sponsor, but will help $______
☐ Please send me further information.

NAME

ADDRESS

CITY STATE ZIP

☐Ck/Money Order ☐Visa ☐Master Card

CARD NO EXP DATE

SIGNATURE

1-800-538-5381

CHILDREN, INC.

U.S. gifts are fully tax deductible.
Annual financial statements are available on request.